Documentation of Software for Phylogenetic Shadowing for Schauer et al.

Version 1.01

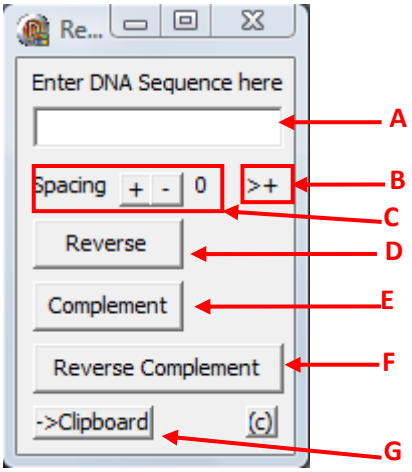
Philipp M. Schlüter, University of Zurich, April 2009

Contact: *philipp.schlueter@systbot.uzh.ch*

This document contains brief documentations for all software made available by the author (PMS) on the following web pages: www.famd.me.uk/agl/agl_sw.html. These pieces of software contain programs used for phylogenetic conservation analysis for the Schauer et al. paper, along with a few small utilities. All programs were written by Philipp M. Schlüter and compiled as executables for Windows operating systems using Delphi 2006.

By downloading and using any of the software, you agree to the licence terms outlined in the legal disclaimer:

Reverse/Complement (RC.exe)



A) Enter sequence in field **A** or paste from clipboard. Sequence can contain ambiguity characters such as Y or N.

B) Field **B** is a 2-character indicator, where ">+" indicates forward, non-complemented sequence. Here, ">" indicates the forward direction, whereas "<" would indicate reverse. "+" indicates a non-complemented and "-" a complemented sequence. These fields are toggled when the respective buttons (**D**,**E**,**F**) are pressed.

C) Use the "+" and "-" buttons to increase/decrease spacing in the sequence. The current spacing is displayed to the right. Sequence "ATTCGG" has spacing 0. With different spacings, it would look like this: 1, "A T T C G G"; 2, "AT TC GG"; 3, "ATT CGG"; etc.

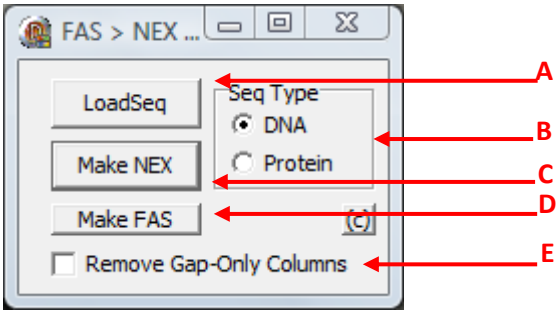
D) Press this button to reverse a sequence. For example, "ATTYCG" will become "GCYTTA".

E) Press this button to complement a sequence. For example, "ATTYCG" will become "TAARGC".

F) Press this button to reverse and complement a sequence. For example, "ATTYCG" will become "CGRAAT".

G) Pressing this button will copy the sequence currently displayed in field **A** onto the clipboard.

.FAS to .NEX Converter (fas2nex.exe)



A) Press the LoadSeq button **A** to display a file open dialogue box and select a FASTA format (typically, *.fas) file to load.

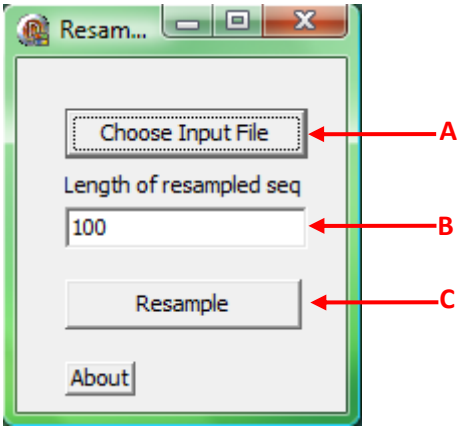
B) Specify whether the sequence loaded is a DNA or protein sequence in field **B**.

C) Press button **C** "Make NEX" to select a file name for the NEXUS (typically, *.nex) file to be written. The program will then immediately create this file. If the sequences in the input file are of unequal length, they will be padded with gap characters, to the length of the longest sequence(s). If option **E** is set, character columns consisting only of gap characters in all sequences will be removed.

D) As an alternative to point **C**, you can press button **D** "Make FAS" to re-save the data in FASTA format. The .fas file written will contain all sequence data in a single line (i.e., there will be no line breaks) and, if option **E** is set, character columns consisting only of gap characters in all sequences will be removed.

E) Tick this checkbox **E** if you do not wish your output file to include gap-only columns when saving a file using buttons **C** or **D**.

Sequence Resampler (ResampleSeq.exe)

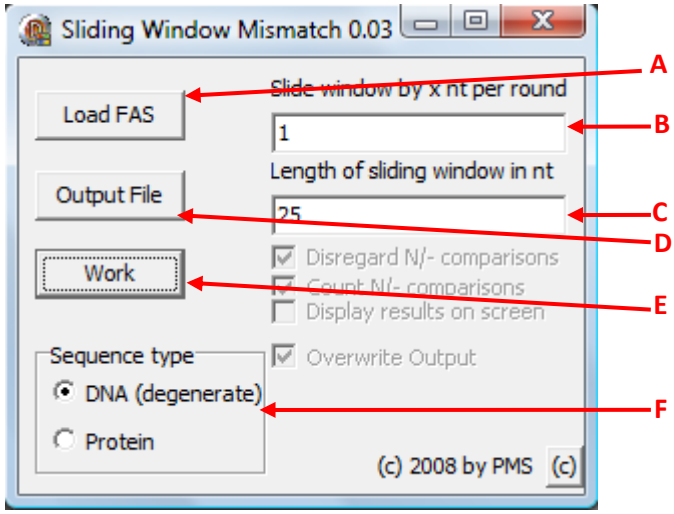


A) Press this button to choose an input file (FASTA format)

B) Indicate the length you want the resampled sequence(s) to have

C) Press this button to perform the resampling. This will bring up a save file dialogue box that lets you choose your output file. The output file format will be in FASTA format.

Sliding Window Mismatch (sdsw.exe)



A) Press button **A** to select a FASTA format input file. Use field **F** to indicate whether input data are DNA or protein.

B) Set the step size of the sliding window in field **B**, i.e., the number of nucleotides a sliding window is shifted relative to the previous window.

C) Set the desired length of the window in field **C**.

D) Press button **D** to specify the desired output file. **NB**: If you select a file which already exists, this file will be overwritten when the analysis (pressing button **E**) is performed. If no file is specified, the default file 'output.txt' will be used.

E) This will perform the sliding mismatch analysis, writing an output file as specified under **D**. For further details of the analysis see below. The output file is a tab-delimited text file that can be opened with any editor or your favourite spreadsheet program.

F) Use this field to indicate whether input data represent protein or DNA sequences.

<u>Sliding mismatch analysis:</u>

A mismatch analysis, calculating a mismatch value in every sliding window for sequences A and B, is performed for every pair of sequences available from the input file, and in addition, the average of all pairwise mismatch values is calculated. This is done in two different ways, designated "S+Gaps" and "S-Gaps". Calculations for S+Gaps and S-Gaps are carried out as follows:

S+Gaps = (SUM (M(i)))/WinSize, summing over all positions i,
where WinSize is the length of the sliding window and M(i) = mismatch at position i.

e.g., position i	M(i)
A/T	1
C/T	1
T/T	0
Y/T	0.5 (probability of mismatch if all nucleotides are equally likely)
-/-	0
N/-	1

S-Gaps = IF G<WinSize THEN (SUM (N(i)))/(WinSize-G) ELSE 0;

where

G = number of pairwise comparisons in which at least one sequence of a given pair contains a gap character (usually, “-“).

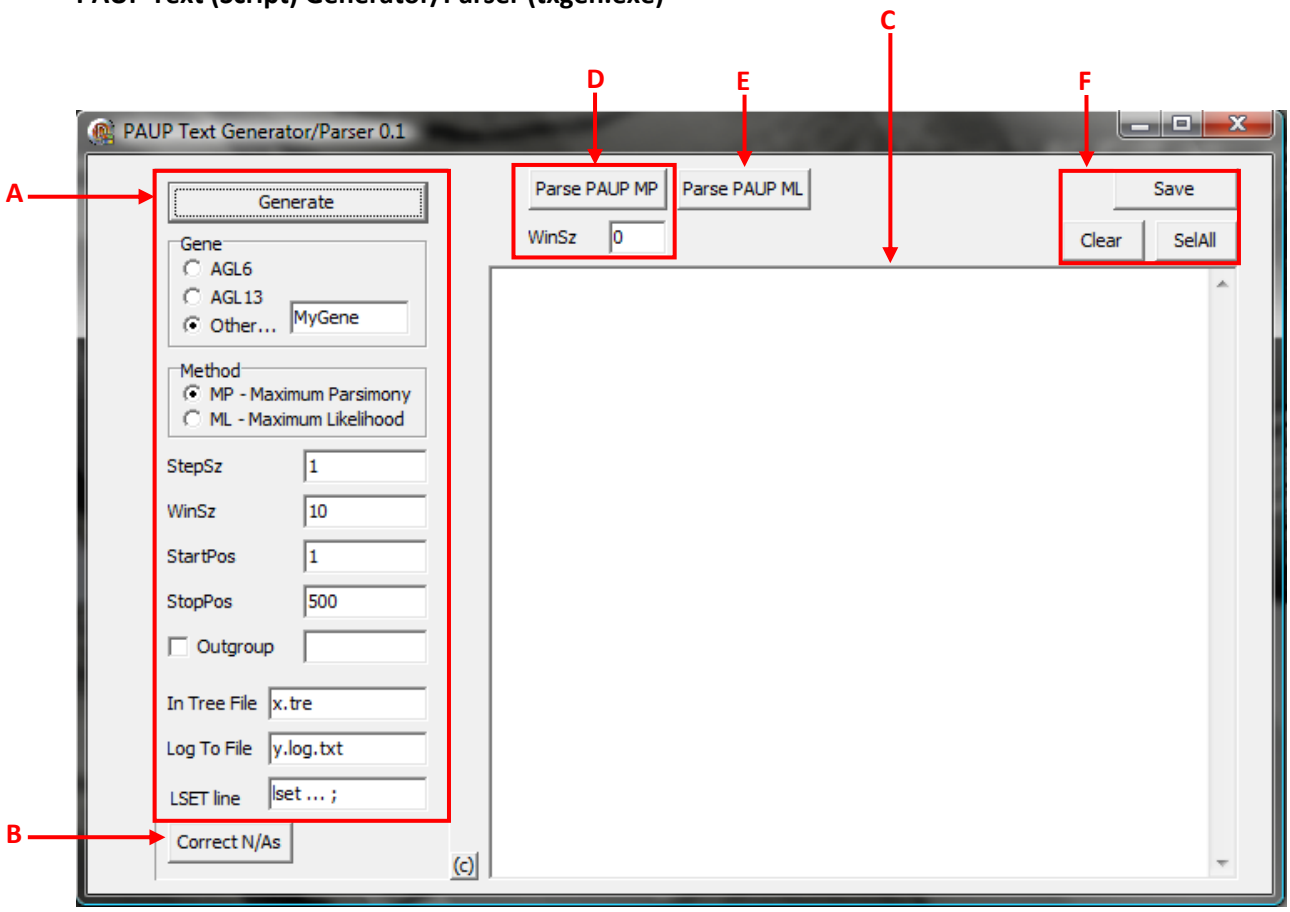
N(i) = IF G(i) = 0 THEN M(i) ELSE 0;

where

G(i) = 1 if there is ≥ 1 Gap Character at position i

Note that S-Gaps is essentially the same as S+Gaps, only getting rid of all comparisons containing a gap character "-".

PAUP Text (Script) Generator/Parser (txgen.exe)



The phylogeny program PAUP*4 can easily be used for calculating the likelihoods or tree lengths of input trees given a section (a window) of input data and it is relatively straightforward to write a script that essentially goes through a set of aligned sequences in a sliding-window fashion. However, writing such a script manually is a little cumbersome. Likewise, retrieving the information provided by PAUP* and storing it in a table amounts to an exceedingly boring, repetitive and even pointless (should I say, existentialistic?) task. Txgen.exe simplifies writing suitable scripts for PAUP* and can parse PAUP*'s output. This program was tested on PAUP*4beta10. To generate a script for PAUP*, use program functions **A** (and, if necessary, **B**). To parse PAUP*'s output file, use functions **D** or **E**. For saving to file, use functions **F**. The script output by this program is in the form of a PAUP command block that can be copied into any regular NEXUS format PAUP* input file, or executed from within PAUP*. See Note 1 for an example script.

NB: See point **B** below if PAUP* displays an error message during execution of the script.

A) Set the parameters in this region according to your needs and then press the "Generate" button to generate a script displayed in area **C**. The "Gene" panel allows you to enter a label for your gene of interest. Use the "Method" selection panel to indicate whether you wish to generate a script for Maximum Parsimony or Maximum Likelihood analysis. The following parameters StepSz, WinSz, StartPos and StopPos indicate the step size, window size, start and stop position along the nucleotide sequence alignment. If you require an outgroup, tick the appropriate checkbox and enter the name of the outgroup sequence in the edit box beside. "In tree file" specifies the name of the input tree file that PAUP* should use as a reference to compare data from every sliding window to. "Log To File" indicates the file you wish PAUP* to write its data to. If doing ML analysis,

enter an appropriate LSET command here to specify the model of molecular evolution to be used.

B) The script generated by this program uses PAUP*'s `describetrees` command to obtain tree scores given a window of data. However, it is possible that for some windows PAUP* may fail to calculate likelihoods (e.g., because not all four nucleotide types are present in that data partition). In this case, PAUP* will fail with an error message and indicate the offending command. In this case, please change the offending "`describetrees;`" line to "`[describetrees;]`" and try again executing the script until it completes (or hits the next error – you may find this a boring exercise, but unfortunately you currently still have to do it manually). Then copy the script back into area **C** of txgen.exe and click button **B**. Doing this will replace a pair of lines like this:

```
[! ###Tree MyGene 2-11]
[describetrees;]
```

by

```
[! ###Tree MyGene 2-11 N/A]
[describetrees;]
```

While you may actually consider this pointless, it isn't. The resulting modified script can ^ now be used in PAUP* AND txgen.exe will be able to interpret PAUP*'s output from it correctly. Note that if PAUP* never complained about anything in the first place (there are no sliding windows for which it cannot compute likelihoods), there is no need to click button **B**.

C) This is the area where a script is displayed or PAUP*'s output should be copied. The contents can further be controlled by functions **F**.

D) Use this to parse a log file created by PAUP* when executing an MP analysis script. You first need to indicate the window size in the WinSz edit box, open the log file in an editor and copy its contents into area **C** and then hit the "Parse PAUP MP" button. The program will then parse the data and ask you for a text file to which to save the parsed data (see Note 2).

E) Use this to parse a log file created by PAUP* when executing an ML analysis script. If PAUP* reported any errors during early tries to execute the script, you may need to follow instructions under point **B** first. Open the log file in an editor and copy its contents into area **C** and then hit the "Parse PAUP ML" button. The program will then parse the data and ask you for a text file to which to save the parsed data.

F) These buttons control region **C**; "Clear" clears the text, SelAll selects the entire contents and "Save" lets you save it to a text file.

Note 1: An example for a script for an ML analysis

```
BEGIN PAUP;
[! ---MyGene ML--- ]
ClearTrees;
GetTrees file=x.tre;
Set criterion=likelihood;
lset Base=(0.2964 0.1469 0.1739) Nst=6 Rmat=(1.00 2.534 1.00 1.00 1.549) Rates=equal Pinvar=0.46;
Outgroup MyOutGroup;

 log file=y.log.txt start=yes replace=yes;

     exclude all;
     include 1-10;
     [! ###Tree MyGene 1-10]
     describetrees;

     exclude all;
     include 2-11;
     [! ###Tree MyGene 2-11]
     describetrees;

     exclude all;
     include 3-12;
     [! ###Tree MyGene 3-12]
     describetrees;

log stop;

END;
```

Note 2: Example of parsed outputs for MP and ML analyses:

MP:

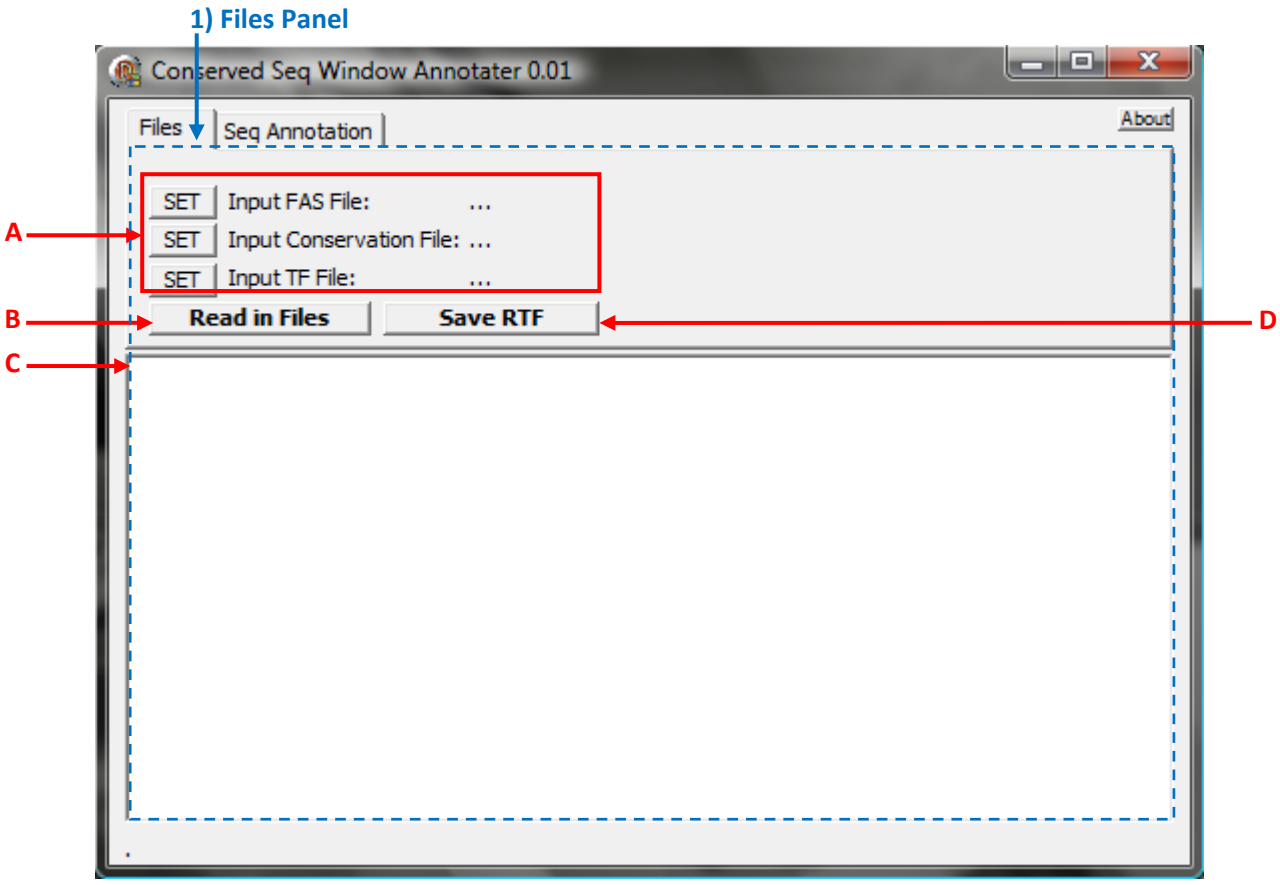
Tree	#ConstChar	#ParsInfChar	TreeLength	CI	HI	CIexclUninf	HIexclUninf	RI	RC
Tree AGL6 1-10	10	0	0						
Tree AGL6 6-15	8	0	2	1.0000	0.0000	0/0	0/0	0/0	0/0
Tree AGL6 11-20	7	0	3	1.0000	0.0000	0/0	0/0	0/0	0/0
Tree AGL6 16-25	6	2	5	0.8000	0.2000	0.6667	0.3333	0.5000	0.4000
Tree AGL6 21-30	5	2	6	0.8333	0.1667	0.6667	0.3333	0.5000	0.4167
Tree AGL6 26-35	6	1	4	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000
Tree AGL6 31-40	6	2	5	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000
Tree AGL6 36-45	6	1	5	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000
Tree AGL6 41-50	6	2	4	1.0000	0.0000	1.0000	0.0000	1.0000	1.0000
Tree AGL6 46-55	8	2	2	1.0000	0.0000	N/A	N/A	1.0000	1.0000
Tree AGL6 51-60	9	0	1	1.0000	0.0000	0/0	0/0	0/0	0/0

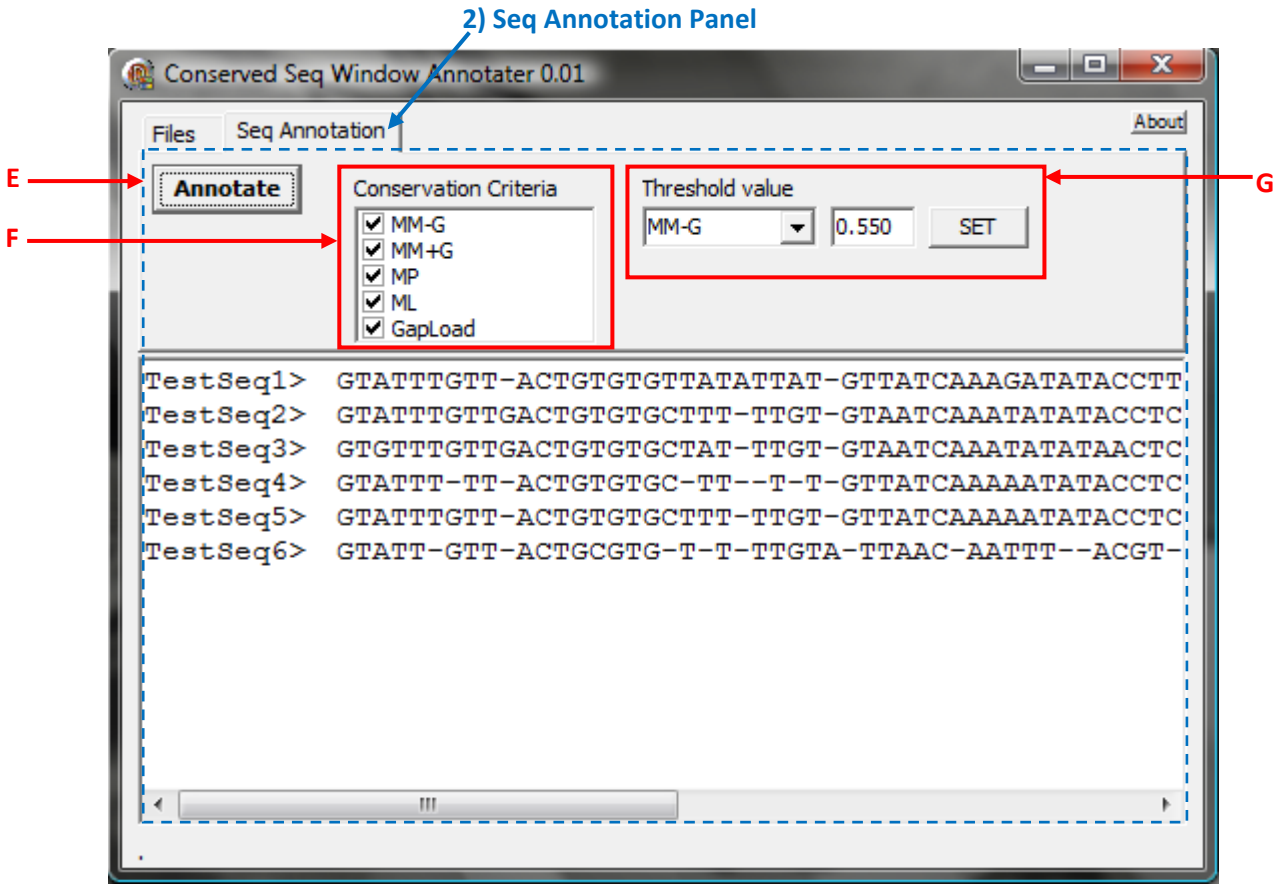
…

ML:

Tree	-ln L
…	
Tree AGL6 6-15	N/A
Tree AGL6 7-16	N/A
Tree AGL6 8-17	22.57251
Tree AGL6 9-18	23.79798
…	

Conserved Sequence Window Annotater (cswA.exe)





This is a tool that lets you visually annotate a set of aligned sequences with information on sequence conservation and transcription factor binding sites. It has two panels, the Files Panel and the Sequence Annotation Panel between which you can alternate.

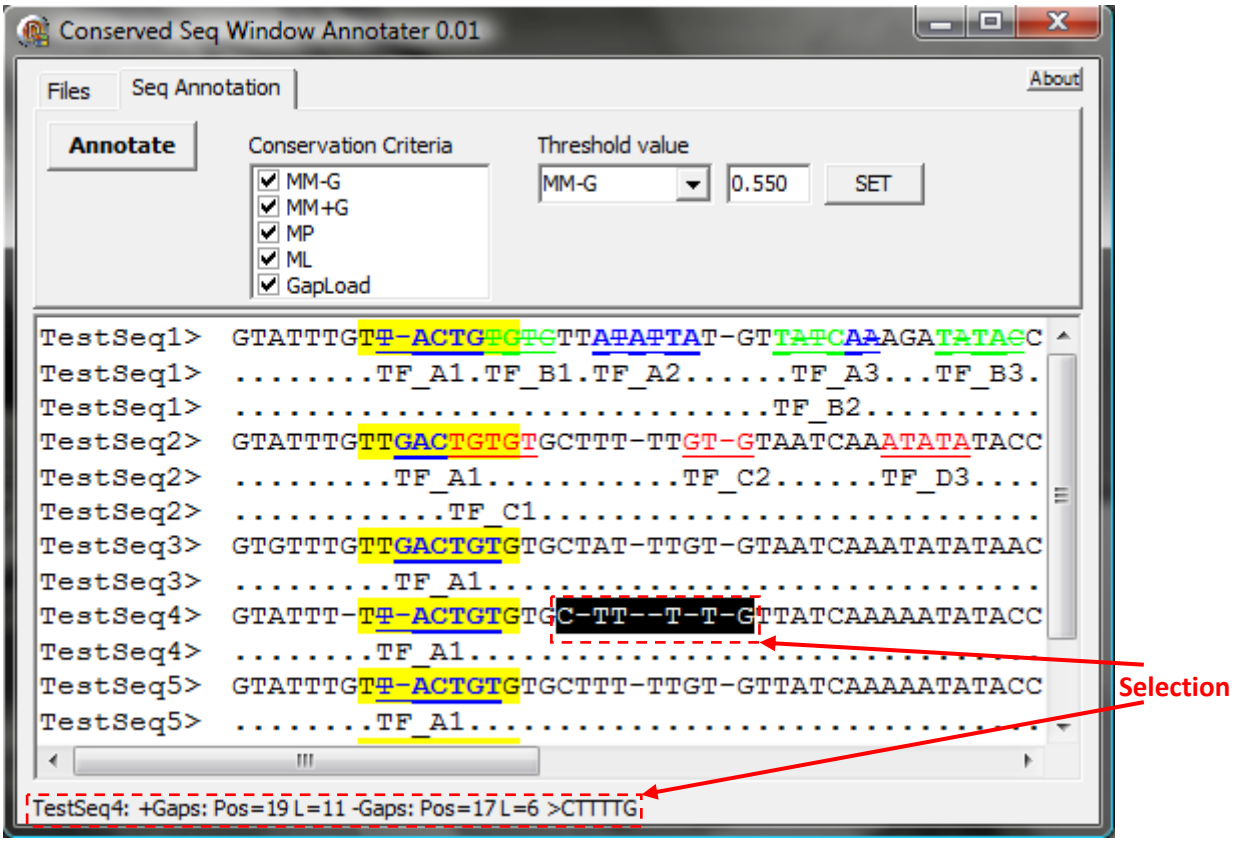
A) Use the "SET" buttons in this area to browse for and select input files for the application. The input FAS file should be a FASTA format sequence file containing aligned sequences. For details on the input files, please see Notes 1-3.

B) Once input files have been selected in **A**, you can press this button to load the files into the application. It will then display the input sequences in area **C**.

C) This area is used to display sequences in either unannotated or annotated format.

D) Clicking this button will save the contents of area **C** to a file in rich text format (*.rtf, see Note 4).

E) Clicking this will annotate the sequence with information from TF and conservation files, where only those conservation criteria selected in area **F** will be considered. Conservation will be highlighted as yellow background colouring, whereas transcription factor binding sites will be mapped onto the input sequences, accounting for gaps in the sequence alignment, and indicated in different foreground colours, e.g., as shown below (every TF set in the TF file will be displayed in a different foreground colour):



If the necessary information is available, TF positions where there is a mismatch among predicted binding sequence (as provided in the TF file) and the actual sequence (as given in the input FAS file)will be indicated as strikethrough-marked text (e.g., ~~C~~), whereas positions of matching sequence will be indicated as bold text (e.g., **A**). Below the line giving the annotated sequence, there will be one or more lines with the TF name (as

given in the TF file) indicated. The first letter of the TF name should be underneath the first nucleotide of the TF binding site. If the names of TFs would otherwise overlap, they will be displayed on several lines.

If text is selected, summary information about the selection will be displayed at the left bottom corner for the (first) sequence under consideration, giving start position and sequence length for both the sequence containing gaps (+Gaps) as well as the ungapped (-Gaps) sequence, and the ungapped selected sequence will be displayed.

NB: If TF binding sites overlap, the colour in the area of overlap will be that of the last TF in the list of TFs in the order of appearance in the TF file.

F) Here you can select which of the possible criteria for considering a sequence conserved (as provided by the conservation file) will be used to annotate the input sequences when button **E** is pressed.

G) This area lets you select a given conservation criterion (as defined in the conservation file)in the combo box and change its associated threshold value by entering a new value into the edit box *and* then pressing the "SET" button. The annotation will not be updated to reflect changes unless button **E** is pressed.

Note 1: Example input FASTA (*.fas) file, with aligned sequences where gaps are indicated by the "-" character:

```
TestSeq1>
GTATTTGTT-ACTGTGTGTTATATTAT-GTTATCAAAGATATACCTTTGATGCCTGA-ATTAGTG-TATAATTTTGTGT-TCTCTTTTTTATTGTCATCC
TestSeq2>
GTATTTGTTGACTGTGTGCTTT-TTGT-GTAATCAAATATATACCTCGGATGCCTGA--T-AGTGTTATAATCTTGTGT-TC-AT-TTTTATTGGCATCC
TestSeq3>
GTGTTTGTTGACTGTGTGCTAT-TTGT-GTAATCAAATATATAACTCGGATGCCTGA--T-AGTGTTATAATCTTGTGTATC-AT-TTTTATTGGCATCC
TestSeq4>
GTATTT-TT-ACTGTGTGC-TT--T-T-GTTATCAAAAATATACCTCGG-TGCCTGA--T-AG-G-TTTA-TCTTGTGT-TC-TT-TTTTGTTG-TGTCC
TestSeq5>
GTATTTGTT-ACTGTGTGCTTT-TTGT-GTTATCAAAAATATACCTCGG-TGCCTGA--T-AG-G-TTTA-TCTTGTGT-TC-AT-TTT-GTTG-TGTCC
TestSeq6>
GTATT-GTT-ACTGCGTG-T-T-TTGTA-TTAAC-AATTT--ACGT-C--TG-G-AGT-T--G-GTTATA-T-TC-TGTAAC-AT-TTT-ATTGTCATCT
```

Note 2: Example Conservation Scores input file. This is a text file that should correspond to the following format:

```
CONSERVATION SCORES FILE
[SCORES]
ScoreNames = MM-G, MM+G, MP, ML, GapLoad;
NSummaryStats = 1;
[[STATS]]
#Stat  MM-G   MM+G   MP     ML     GapLoad
Thresh 0.55   0.6    11     12     15
[[TABLE]]
#Start Stop   MM-G   MM+G   MP     ML     GapLoad
8      16     0.2    0.3    10     11.2   14.2
20     28     0.45   0.1    15     15     20
30     38     0.8    0.7    1      14.4   20
40     48     0.8    0.8    70     6.2    22
50     58     0.8    0.8    70     N/A    27
60     68     0.2    0.2    6      6.5    0
*
```

Here, the names (highlighted in red above) for the conservation criteria are defined as a comma-separated and semicolon-terminated list after the "`ScoreNames =`" variable definition. Names should not contain blanks etc. The exact names you enter here will be displayed in area **F** of the program. Initial threshold values (green) *below* which a given region of sequence may be considered conserved under

the given criterion are provided as tab-delimited values. Finally, a table of candidate areas of conservation (as derived from e.g., phylogenetic shadowing) is provided as tab delimited values (black), including start and stop positions along the sequence alignment and observed scores for the different conservation criteria/score types provided earlier (where "N/A" is a possible input value leading to this particular score instance to not be considered for assessing overall conservation criteria). Please note that the end of the table is marked by a "*" character.

Note 3: Example Transcription Factor Binding Sites Annotation (TF) File:

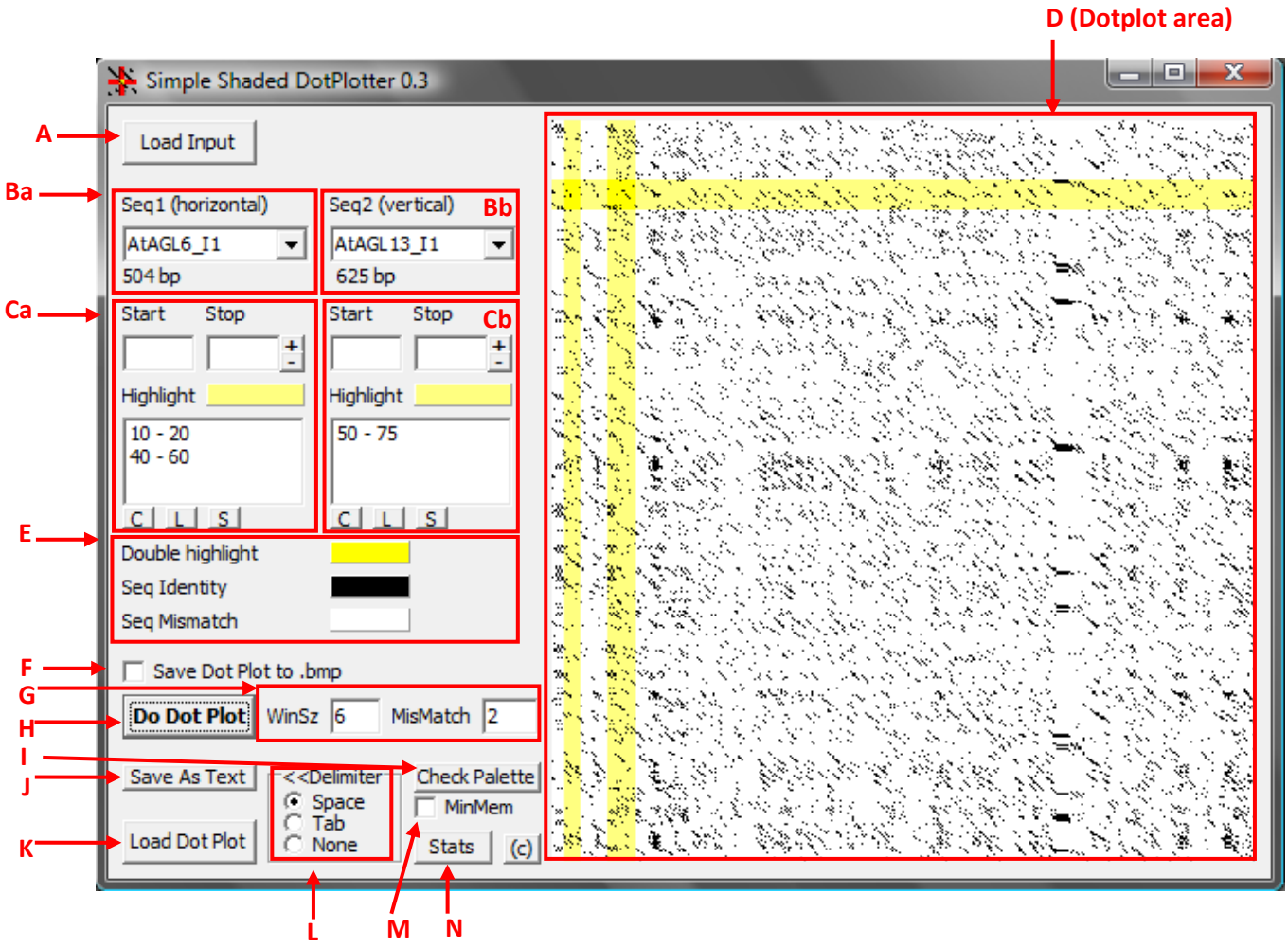
```
BINDING SITES FILE
[TF]
Name = TF1;
MatchType = direct;
[[TABLE]]
#RefSeq      Start  Name   Seq
TestSeq1     9      TF_A1  GACTGT
TestSeq1     20     TF_A2  ACACTA
TestSeq1     30     TF_A3  ATCAT
TestSeq6     28     TF_A4  CAAT
TestSeq2     10     TF_A1  GACTGT
TestSeq3     10     TF_A1  GACTGT
TestSeq4     8      TF_A1  GACTGT
TestSeq5     9      TF_A1  GACTGT
TestSeq6     8      TF_A1  GACTGT
*
[TF]
Name = TF2;
MatchType = case-sensitive;
[[TABLE]]
#RefSeq      Start  Name   Seq
TestSeq1     12     TF_B1  tcGGATgc
TestSeq1     25     TF_B2  tatcTTACaat
TestSeq1     35     TF_B3  tatTTTAGaa
TestSeq1     57     TF_B4  taAGTCaa
*
[TF]
Name = TF3;
MatchType = length;
[[TABLE]]
#RefSeq      Start  Name   Length
TestSeq2     13     TF_C1  5
TestSeq2     25     TF_C2  3
TestSeq2     35     TF_D3  5
TestSeq2     55     TF_D4  6
*
```

This example contains three sets of TF binding sites, which are declared after the "`Name =`" (highlighted in red, TF1, TF2, TF3). For every set, a there is a `MatchType` variable and a table (black above) containing the information the program can use for sequence annotation. All tables are terminated with an asterisk ("`*`"character). The program expects different information in the table depending on which match type was chosen. Currently, cswA.exe can deal with three different match types called "`direct`", "`case-sensitive`" and "`length`" (highlighted green above). In every case the first column lists the reference sequence (one of the input sequences) that is to be annotated, the start position (position of the first nucleotide of the binding site) in that sequence (not counting any gap characters) and the name the program will use to label the site in question. In "`length`" mode, a fourth column is provided that gives the length of the binding site. In "`direct`" mode, the expected/consensus sequence is listed in the fourth column. In this case, the program can try to highlight match and mismatch to this sequence. Like in "`direct`" mode, in "`case-sensitive`" mode a sequence is provided in the fourth column, where the position indicated in the second column refers to the first nucleotide of the sequence provided in the fourth column, however, the program will only use *uppercase* letters of the fourth-column sequence for highlighting match or mismatch.

Note 4: Example output Rich Text Format (*.rtf) file:

```
TestSeq1> GTATTTGTT-ACTGTGTGTTATATTTAT-GTTATCAAAGATATACCTTTGATGCCTGA-ATTAGTG-TATAATTTTGTGT-TCTCTTTTTTATTGTCATCC
TestSeq1> ........TF_A1.TF_B1.TF_A2......TF_A3...TF_B3.................TF_B4.............................
TestSeq1> ..............................TF_B2.....................................................................
TestSeq2> GTATTTGTTGACTGTGTGCTTT-TTGT-GTAATCAAATATATACCTCGGATGCCTGA--T-AGTGTTATAATCTTGTGT-TC-AT-TTTTATTGGCATCC
TestSeq2> .........TF_A1...........TF_C2......TF_D3..............TF_D4.....................................
TestSeq2> ............TF_C1.....................................................................................
TestSeq3> GTGTTTGTTGACTGTGTGCTAT-TTGT-GTAATCAAATATATAACTCGGATGCCTGA--T-AGTGTTATAATCTTGTGTATC-AT-TTTTATTGGCATCC
TestSeq3> .........TF_A1.....................................................................................
TestSeq4> GTATTT-TT-ACTGTGTGC-TT--T-T-GTTATCAAAAATATACCTCGG-TGCCTGA--T-AG-G-TTTA-TCTTGTGT-TC-TT-TTTTGTTG-TGTCC
TestSeq4> ........TF_A1.....................................................................................
TestSeq5> GTATTTGTT-ACTGTGTGCTTT-TTGT-GTTATCAAAAATATACCTCGG-TGCCTGA--T-AG-G-TTTA-TCTTGTGT-TC-AT-TTT-GTTG-TGTCC
TestSeq5> ........TF_A1.....................................................................................
TestSeq6> GTATT-GTT-ACTGCGTG-T-T-TTGTA-TTAAC-AATTT--ACGT-C--TG-G-AGT-T--G-GTTATA-T-TC-TGTAAC-AT-TTT-ATTGTCATCT
TestSeq6> ........TF_A1...................TF_A4.....................................................................
```

Dotplotter (dotplot.exe)



The main functions of this program are to (1) generate dotplots, highlighting/shading regions in the plot and (2) to evaluate these, outputting numerical information about dotplots. These two functions can be used independently of each other. For function (1), start with button **A**, and carry out dotplot analysis via button **H**. For function (2), start as in (1) or alternatively load a bitmap (*.bmp) file previously generated with this program via button **K**, and then generate basic stats for the dotplot by pressing button **N**.

A) Press this button to load a set of sequences (FASTA format) whose names will be listed in both areas **Ba** and **Bb**.

B) Use these combo-boxes to select a pair of sequences for dotplot analysis, where Seq1 will be displayed horizontally and Seq2 vertically on a given dotplot. The length of a selected sequence in bp will be indicated below the combo-box.

C) Use fields **Ca** and **Cb** to select regions along sequences Seq1 and Seq2 (see **B**), respectively, to be highlighted. Multiple regions can be entered for each sequence. To enter a region, enter the start and stop positions (both of which will be included in your region) in the respective edit boxes and press the small "**+**" button. The region will then be listed in the list box below. Selecting a region in this list box (by mouse click) and then hitting the "**-**" button will delete this region from the annotation list. The "**C**", "**L**" and "**S**" buttons below the list box can be used to clear the box, load or save its contents to a text file (see note 1). Clicking on the coloured "Highlight" panel lets you select a background colour in which the defined regions will be displayed on the dotplot.

NB: Please note that regions are defined in terms of *pixels* along the dotplot, *not in bp*. Therefore, if you want to define a region that ends with the last nucleotide of an input

sequence, it needs to end at position P=L-W+1, where L is the length of the sequence (as displayed in area **B**) and W is the Window Size (as set in region **G**). If a region is not validly defined (e.g. if its start and/or stop positions are outside the pixel positions of the resulting dotplot), it will *not* be displayed on the plot! Please also note that selecting a different sequence in area **B** does *not* change the list of regions to be highlighted.

D) This area holds the dotplot (which will usually be scaled to fit the area available).

E) Click on the coloured panels in this area to select the colours used for the dotplot. The double highlight colour defines how the intersections of regions defined in **Ca** (horizontal) and **Cb** (vertical) will be displayed (the background colour for the intersection areas). Seq Identity defines the colour in which a dot for a matching stretch of sequences will be displayed on the plot (foreground colour) and Seq Mismatch defines the colour to be displayed on the plot if a pair of sequence windows does not match if it does not lie in a highlighted region (default background colour).

F) Check this checkbox if you wish the dotplot to be saved to file when it is generated (button **H**).

G) Set the window size (WinSz) and allowed MisMatch per sequence window for the dotplot procedure (button **H**) here.

H) Press this button to generate a dotplot in area **D** from the pair of sequences selected in **B** and parameters defined in **G** and the highlighting parameters in **C** and **E**. If checkbox **F** is ticked, the program will display a save file dialogue box to allow you to save the dotplot to file (*.bmp, see Note 2). The dotplot procedure will go through every possible pair of windows of length WinSz (defined in **G**) among the selected sequences. If the amount of pairwise sequence mismatch in this window is less than or equal to the MisMatch value defined in **G**, a pixel will be drawn in the Seq Identity colour specified in **E**. Otherwise, a pixel will be drawn in the Seq mismatch colour (specified in **E**) or, in the appropriate highlight or double-highlight colour (specified in **C** and **E**).
NB: Currently, only a step size of 1 is implemented for dotplots.

I) Clicking this button will display the current bitmap's internal colour palette. This feature will probably be of little interest to most users, but may be useful if you wish to write your own image analysis code or manipulate the images manually.

J) Clicking this button will display a save file dialogue and then save the dotpot as a text file. In this file, individual pixels from the dotplot will be displayed as characters, where "0" indicates a window position of sequence identity and "1" – "4" indicate sequence mismatch. Here, "1" is the default mismatch character, whereas "2" and "3" indicate sequence mismatch in a region highlighted on sequences 1 (horizontal) or 2 (vertical), respectively. "4" indicates sequence mismatch in a double-highlight region. You can use the radio buttons in area **L** to define if (how) characters in this file should be delimited.

K) Use this button to load an 8-bit (*not* a 24-bit) bitmap (*.bmp, see Note 2) from file to be displayed in area **D**. Generally, the analysis of this bitmap (e.g., point **N**) will only be possible if this bitmap was created by this program.

L) Use this group of radio buttons to indicate which character delimiter should be used when exporting a dotplot to text using button **J**. You can either use a space or tab character, or no delimiter at all.

M) Checking this option will reduce the peak memory consumption during the numerical analysis of a dotplot (button **N**), which may make sense if the dotplot is large. The program will free the memory of the bitmap (the display in **D** will be cleared).

N) Clicking this button will first ask for an output file and then extract some basic numerical information from the dotplot and save it to the output file. For large dotplots, option **M** can be checked to reduce memory consumption and allow the analysis of somewhat larger bitmaps. Extraction of numerical information can be carried out either or a dotplot previously generated by a click on button **H**, or on a dotplot saved by the program during a previous session and loaded by a click on button **K**. If a sequence input file was loaded (button **A**) and the dimensions of the current dotplot match those expected from the

pair of sequences selected in **B** and the WinSz parameter in **H**, the program will ask whether the dotplot represents the pair of sequences in **B**. If so, the program will be able to output sequence information for matching sliding windows. In every case, the program will output the size and starting position for each diagonal (top left to bottom right) or individual pixel indicating a sequence match, along with its starting position and the number of its pixels that lie in any of the highlighted regions. Also, there will be the areas of the regions under consideration and two summary tables on the occurrence of diagonals of different size classes (for every highlighted region), either in absolute values or scaled by the respective areas. For further details, please see Note 3.
NB: Please note that on most Windows systems, there will still be a memory limit the program can hit when trying to analyse very large bitmaps. This is not a bug in this software, but due to the operating system which will eventually deny giving the program more memory. In such cases an "Out of memory" error will occur. Should the program still produce an output file in this case, please discard it, as it cannot be expected to be reliable.

Note 1: A text file defining regions for highlight will correspond exactly to the following format:

```
1 - 5
50 - 50
90 - 95
```

Note 2: The program writes and reads 8-bit bitmap (*.bmp) files instead of 24-bit ones, because a dotplot will only require a very modest number of colours and 8-bit bitmaps offer a 4-fold reduction in file size over standard 24-bit bitmaps, which is noticeable for longer input sequences. (Yes, in principle, more compressed 4-bit bitmaps could have been used but these would have been more cumbersome to implement and may under some circumstances fail to preserve the colour information embedded in them.) 8-bit bitmaps contain a palette that store the 24-bit colour information and when reading bitmap files the program requires the bitmap's internal palette to be in the same way it was written to file by the program (i.e., don't manipulate it or convert the file to a different image format if you want to be able to read it back into this program).

Note 3: The output file (from point **N**) gives a numerical summary of the dotplot. The file is tab-delimited and should thus be easy to import in your favourite spreadsheet program. The file refers to X and Y coordinates of the dotplot, where X corresponds to the first input sequence (horizontal) and Y to the second one (vertical). If sequence information was available with the dotplot, the sequence names for X and Y will be indicated.

The first section of the output file contains a list of actual diagonals (or single pixel presences) found in the dotplot, along with their start coordinates (StartX and StartY) and their length in pixels. If highlighted regions were found in the dotplot, the number of pixels of a given diagonal that are located in any of the highlighted regions (or intersection of highlighted regions along the X and Y axes) are noted. Highlighted regions are indicated as e.g. X10-20 for a highlighted region along the X axis including positions X=10 to X=20. (The leftmost coordinate of the dotplot is here referred to as X=1, Y=1), or as X10-20xY20-30 for a highlighted intersection area. If sequence information was available for analysis of the dotplot, the consensus sequence and the sequences of the two participating inputs are also noted under the SeqCons, SeqX and SeqY columns.

The second section of the output file gives summary values for the diagonals listed in section 1. Summary values are given for every diagonal size class, where N indicates the total number of diagonals of a given size class that were found. The remaining values indicate the sum of diagonal fractions present in any of the highlighted regions or region intersections. This means that if two pixels of a diagonal of length of 10 pixels lie in a highlighted region, the corresponding diagonal fraction will be 2/10=0.2. The columns AllX, AllY, AllXY (if present) indicate the *sum* of diagonal fractions in *all* highlighted regions of X, Y or in X x Y intersections. Likewise, the columns NonX, NonY, NonXY indicate the sum of diagonal fractions along the indicated axes that are *not* in any highlighted region.

The third section lists the areas (in pixel²) that for every region listed in section 2, where the area in column N is the total area of the dotplot.

The fourth section of the output file has the same structure as section 2, but instead of absolute diagonal counts, lists the scaled size class values ("diagonal densities"). These are the values given in section 2 divided by the areas given in section 3.

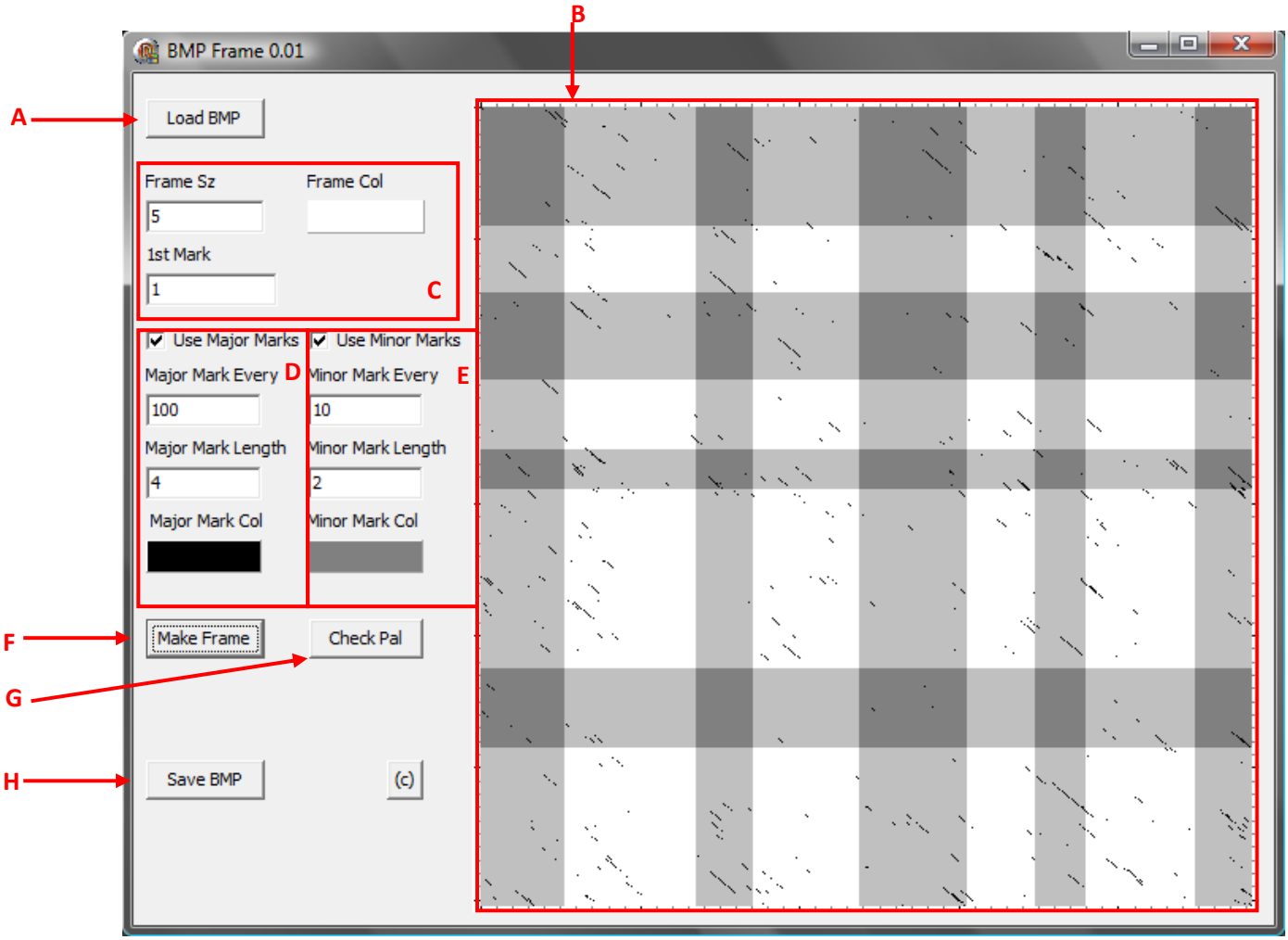
Example (sections of an) output file are provided below:

```
X= 495 pixels: YourSeq1
Y= 616 pixels: YourSeq2
Number  StartX  StartY  Length  X20-50  Y30-60  XY#x#   SeqCons         SeqX            SeqY
1       486     15      2       0       0       0       wTTmATTsAAk     TTTAATTGAAT     ATTCATTCAAG
2       487     30      3       0       3       0       TTAATTsrrTTT    TTAATTGAATTT    TTAATTCGGTTT
3       478     32      3       0       3       0       AwTTykGTTTTm    ATTTTTGTTTTA    AATTCGGTTTTC
...
Size    N       NonX    AllX    X20-50  NonY    AllY    Y30-60  NonXY   AllXY   XY20-50x30-60
Size Classes:
1       607     571.000 36.000  36.000  589.000 18.000  18.000  607.000 0.000   0.000
2       285     272.500 12.500  12.500  272.000 13.000  13.000  284.000 1.000   1.000
3       153     143.667 9.333   9.333   150.000 3.000   3.000   153.000 0.000   0.000
4       96      86.000  10.000  10.000  90.500  5.500   5.500   96.000  0.000   0.000
5       75      68.200  6.800   6.800   70.800  4.200   4.200   75.000  0.000   0.000
6       47      44.000  3.000   3.000   46.000  1.000   1.000   47.000  0.000   0.000
7       32      28.429  3.571   3.571   30.000  2.000   2.000   32.000  0.000   0.000
8       14      14.000  0.000   0.000   10.750  3.250   3.250   14.000  0.000   0.000
9       11      9.444   1.556   1.556   11.000  0.000   0.000   11.000  0.000   0.000

Areas:
        304920  285824  19096   19096   289575  15345   15345   303959  961     961

Scaled Size Class Values:
1       0.001990686082  0.001997732871  0.001885211563  0.001885211563  0.002034015367  0.001173020528 ...
2       0.000934671389  0.000953383901  0.000654587348  0.000654587348  0.000939307606  0.000847181492 ...
3       0.000501770956  0.000502640319  0.000488758553  0.000488758553  0.000518000518  0.000195503421 ...
4       0.000314836678  0.000300884460  0.000523669879  0.000523669879  0.000312526979  0.000358422939 ...
5       0.000245966155  0.000238608374  0.000356095517  0.000356095517  0.000244496244  0.000273704790 ...
6       0.000154138791  0.000153940887  0.000157100964  0.000157100964  0.000158853492  0.000065167807 ...
7       0.000104945559  0.000099461807  0.000187024957  0.000187024957  0.000103600104  0.000130335614 ...
8       0.000045913682  0.000048981191  0.000000000000  0.000000000000  0.000037123370  0.000211795373 ...
9       0.000036075036  0.000033042867  0.000081459759  0.000081459759  0.000037986705  0.000000000000 ...
10      0.000022956841  0.000024490596  0.000000000000  0.000000000000  0.000020720021  0.000065167807 ...
11      0.000016397744  0.000017493283  0.000000000000  0.000000000000  0.000017266684  0.000000000000 ...
12      0.000006559097  0.000006997313  0.000000000000  0.000000000000  0.000006906674  0.000000000000 ...
13      0.000000000000  0.000000000000  0.000000000000  0.000000000000  0.000000000000  0.000000000000 ...
14      0.000003279549  0.000003498657  0.000000000000  0.000000000000  0.000003453337  0.000000000000 ...
```

BMP Frame (BmpFrame.exe)



This program allows to load an 8-bit bitmap file (*.bmp, e.g., one created by dotplot.exe) draw a frame with tick marks around it and re-save it as an 8-bit bitmap file.

NB: Once a frame has been created around a bitmap using this program, the resulting bitmap *cannot* be used for creating a numerical summary by dotplot.exe. (It technically can, but the results of this analysis would *not* correctly reflect the data contained in the dotplot.)

A) Use this button to load an 8-bit bitmap file (*.bmp) which will be displayed in area **B**. If the bitmap is larger than area B in terms of pixels, it may not be displayed correctly in area **B**, however, this does not impair program function, because the bitmap will still be kept in memory correctly.

B) The loaded bitmap (with or without frame) will be displayed in this area.

C) Set overall parameters for the frame to be generated here. The “Frame Sz” value indicates the width (in pixels) of the frame to be drawn around the input image. Clicking on the “Frame Col” button lets you select a background colour for the frame. The “1st Mark” value determines the position (in pixels) along every axis at which the first tick mark should be drawn. The top left pixel of the bitmap has coordinates (1,1).

D) Indicate whether to draw major tick marks by checking/unchecking the “Use Major Marks” checkbox. The “Major Mark Every” field lets you specify the spacing (in pixels) of tick marks to be drawn. Further, the length (in pixels) of tick marks can be indicated; they will always be drawn at a width of 1 pixel. Clicking on the “Major Mark Col” panel lets you select the colour in which the marks will be drawn.

E) Like **D**, but for minor tick marks. If a minor and major tick mark is drawn at the same position, the major mark will be drawn over (i.e., will overwrite) the minor mark.

F) Click here for the program to generate a frame around the input image using the parameters specified in **C**, **D** and **E**.

G) Clicking this button will display the bitmap's internal 8-bit palette.

H) Click this button to save the current 8-bit bitmap to file.